IMPLEMENTASI ALGORITMA GREEDY STRING TILING PADA PENDETEKSIAN KEMIRIPAN PROGRAM JAVA

Sulistyowati[1], Hedrianto[2], Andy Rachman[3]
Jurusan Teknik Informatika – Institut Teknologi Adhi Tama Surabaya[1]
Jurusan Teknik Informatika – Institut Teknologi Adhi Tama Surabaya[2]
Jurusan Teknik Informatika – Institut Teknologi Adhi Tama Surabaya[3]
e-mail: aryis77@gmail.com[1],andy.rach1910@itats.ac.id[3]

ABSTRACT

In the era of globalization, people are required to create and innovate to create a work. But unfortunately there are still many people who do negative actions by plagiarizing ideas or thoughts of others. This is also often the case in the world of programming, where many programs are even logically similar or similar to others. In this study, researchers focused on the process of detecting similarities Java programs. Similarity checking process in this study using Greedy String Tiling algorithm. The Greedy String Tiling algorithm compares two data inputs in which the algorithm compares each row in the program. This algorithm has two testing phases: phase before and phase after process. Test applications performed as much as 1200 times on 20 files with different data lengths. From the test results it was found that in length 1 yielded 10% weight, 72% and light weight 18%. At length 2 we found the similarity of weight 8%, moderate 25% and light 67%. In length 3 we get the similarity of weight 7%, 13% and light 80%.

Keywords: programming, Java program, Greedy String Tiling , similarity.

ABSTRAK

Di era globalisasi saat ini, orang dituntut untuk berkreasi dan berinovasi menciptakan suatu karya. Namun sayangnya masih banyak orang yang melakukan tindakan negatif dengan menjiplak ide atau pikiran orang lain. Hal ini juga sering terjadi di dunia pemrograman, dimana banyak beberapa program yang struktur bahkan logikanya sama atau mirip dengan orang lain. Pada penelitian ini, peneliti berfokus pada proses pendeteksian kemiripan program Java. Proses pengecekan kemiiripan pada penelitian ini menggunakan algoritma Greedy String Tiling . Algoritma Greedy String Tiling membandingkan dua inputan data dimana algortima ini membandingkan setiap baris pada program. Algortima ini memiliki dua fase pengujian yaitu fase sebelum dan fase sesudah proses. Uji aplikasi dilakukan sebanyak 1200 kali pada 20 file dengan panjang data yang berbeda. Dari hasi uji didapatkan bahwa pada length 1 menghasilkan kemiripan berat 10%, sedang 72% dan ringan 18%. Pada length 2 didapatkan kemiripan berat 8%, sedang 25% dan ringan 67%. Pada length 3 didapatkan hasil kemiripan berat 7%, sedang 13% dan ringan 80%.

Kata Kunci : *pemrograman, program Java, Greedy String Tiling , kemiripan.*

PENDAHULUAN

Teknologi informasi dan komunikasi saat ini sudah sangat berkembang di Indonesia, mulai dari perkembangan dibidang ekonomi sampai dengan perdangan bahkan sampai dunia pendidikan. Selain sisi positif perkembangan teknologi informasi dan komunikasi juga memberikan sisi negatif. Salah satu sisi negatif yang dialami ada pada bidang pendidikan. Pada dunia pendidikan kegiatan plagiarisme ini terjadi dari sisi pelajar, mahasiswa bahkan sampai dengan dosen[1]. Plagiarisme merupakan tindakah pencurian kekayaan intelektual. Plagiarisme adalah suatu pengakuan terhadap karya orang lain menjadi miliknya sendiri, pengakuan tersebut mulai dari ide, bahasa, cara, produk ataupun catatan atau tulisan orang lain tanpa sepengetahuan pemilik aslinya[2].

Perangkat lunak merupakan salah satu bagian dari sistem komputer. Perangkat lunak dibagi menjadi dua bagian besar, perangkat lunak sistem dan perangkat lunak aplikasi. Perangkat lunak aplikasi merupakan perangkat lunak yang diciptakan untuk menyelesaikan masalah

pengguna[3]. Aplikasi modern saat ini lebih kompleks bila dibandingkan dengan aplikasi waktu lampau. Kekompleksan aplikasi ini dengan tujuan untuk memberikan pelayanan dan kualitas yang lebih bagi penggunanya[4]. Untuk membuat sebuah perangkat lunak, pengembang harus menuliskan perintah-perintah yang terurut dan mempunyai maksud unjuk kerja tertentu[5]. Perintah-perintah yang dituliskan oleh pengembang agar dapat berjalan diperlukan bahasa pemrograman tertentu. Bahasa Pemrograman digunakan untuk mempermudah mengembangkan algoritma yang diinginkan oleh pengembang[6]. Bahasa pemrograman memiliki jumlah yang sangat banyak, tetapi dari sisi penggunaan terdapat lima bahasa pemrograman yang popular digunakan yaitu java – 23.1%, Python – 14.4%, PHP – 9.7%, C# – 8.4% dan JavaScript – 7.7% [7]. Bahasa pemrograman java juga masih banyak digunakan pada jurusan teknik informatika sebagai salah satu mata kuliah pemrograman.

Plagiarisme dibangku perkuliahan masih terjadi sangat besar, terutama di jurusan teknik informatika. Plagiasi kode sumber program sering dilakaukan siswa dengan sangat mudah meniru pekerjaan temannya, memodifikasinya, dan mensubmit layaknya pekerjaannya sendiri. Plagiarisme dapat dilakukan oleh setiap siswa bahkan yang mengetahui pemrograman dengan baikpun juga melakukan plagiarisme[8]. Algoritma Greedy String Tiling merupakan suatu algoritma pencocokan string (kalimat)[9]. Algoritma ini akan melakukan pencarian dua kalimat untuk menemukan bagian umum dari kalimat yang ada[10]. Fokus penelitian ini adalah dekteksi kemiripan program java yang dilakukan dikalangan mahasiswa dengan menggunakan algoritma Greedy String Tiling .

TINJAUAN PUSTAKA

Greedy String Tiling

Greedy String Tiling merupakan algoritma yang digunakan untuk proses pendeteksian plagiasi. Pada saat algoritma GST digunakan untuk membandingkan dua string token, tujuannya untuk menemukan satu set substring yang sama pada kedua file kode program. Setiap tanda yang ada pada file kode program pertama akan dicocokan dengan beberapa token yang sama, yang ada pada file kedua kode program. Setiap tanda pada garis kode sumber dicocokan dengan salah satu tanda dari baris kode sumber kedua. Substring dapat ditemukan tergantung pada posisi mereka dalam string, tetapi posisi relative tidak berubah[9]. Greedy String Tiling ini termasuk pada algoritma fitur teks atau algoritma berbasiskan teks[15].

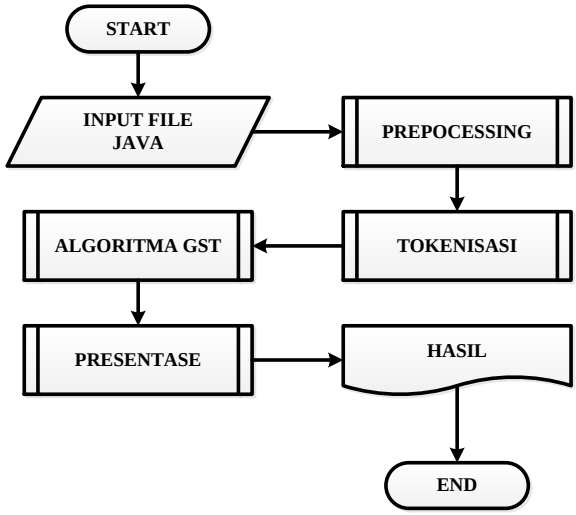
Algoritma GST memiliki dua fase tahap dalam proses, sebelum menuju pada fase tahap pertama dengan melakukan langkah preprocessing yaitu spasi (*white space*), komentar dan string dihapus[16]. Statement pada Bahasa yang tersisakan akan digantikan oleh Token. Pada fase pertama, kedua file kode program yang dicari dan dibandingkan secara berurutan, agar bisa mencapai tujuan ini semua token yang ada pada file kode program yang pertama akan dibandingkan dengan setiap tanda pada file kode program yang kedua. Jika token *identic* (cocok) maka substring akan dikumpulkan. Pada fase kedua, semua perbandingan yang ada pada tahap pertama akan ditandai, Setiap token hanya dapat digunakan dalam satu kali pencocokan. Jika digunakan lebih dari satu kecocokan akan mengakibatkan overlap. Proses ini akan diulang sampai tidak ada pencocokan yang ditemukan. Minimum match length digunakan menghindari kesalahan kecocokan pada algoritma GST untuk mendeteksi kemiripan.

```
GreedyST(String Nilai1, String Nilai2) {
   Tiles = {};
   Do {
      Maxmatch =MinimumMatchLenght;
      Matches = {};
      Forall unmarked tokens Aa in Nilai1 {
         Forall unmarked tokens Nilai2b in Nilai2 {
            J=0;
```

```
            While    (Nilai1a+j    ==    Nilai2b+j    &&unmarked(Nilai1a+j)    &&
                 unmarked(Nilai2b+j))
            J++;
            If(j == maxmatch)
               Matches = matches + match(a,b,j)};
               Maxmatch = j;
            }
         }
      }
      Forall match (a, b, maxmatch) in matches {
         For j = 0.. (maxmatch -1) {
            Mark(Nilai1a+j);
            Mark(Nilai2b+j);
         }
      Tiles = tiles + match (a,b,maxmatch);
Return tiles;
```

METODE

Proses pengecekkan atau pendeteksian kemiripan program java yang dilakukan oleh peneliti menggunakan empat kegiatan utama, yaitu prepocessing, tokenisasi, implementasi algoritma greedy string tiling dan terakhir adalah menghitung presentasi kesamaan dua progra java. Proses preporcessing digunakan untuk menghapus : string, spasi, dan komentar program.



Gambar 1. Proses Deteksi Plagiarime Program Java

Pada gambar 1 didapatkan bahwa proses Algoritma GST dilakukan setelah proses prepocessing dan tokensisasi. Proses prepocessing akan melakukan penghapusan string pada sebuah program. String pada sebuah program java ditandai dengan tanda petik ganda misalnya : **`system.out.println("String atau Kalimat");`**, maka hasilnya berupa **`system.out.println("");`**. Langkah kedua dari proses preprocessing adalah menghapus spasi pada program java. Langkah ketiga dari proses preprocessing adalah menghapus semua komentar program yang ada pada program java, dimana komentar pada program java ditandai dengan tanda double slash (//) sebagai komentar tunggal dan komentar yang diawali dengan tanda slash bintang (/*) dan diakhiri dengan bintang slash (*/).

Tabel 1. Proses Preprocessing Program Java

SEBELUM PREPROCESSING	SESUDAH PREPROCESSING

/*
* Aplikasi Pembantu Menghitung Deret Fibonacci
* Data deret maksimal 20
* Di Create Tanggal : 2016
* Di Surabaya
*/
 	
public class FiboFibonacci { //Class Fibanacci
 	
public static void main(String[] args) {
 	
//Data Fibonacci Maksimal
int Batas = 20;
 	
long[] series = new long[Batas];
 	
//create first 2 series elements
series[0] = 0;
series[1] = 1;
 	
//create the Fibonacci series and store it in an array
for(int i=2; i < Batas; i++){
series[i] = series[i-1] + series[i-2];
}
 	
//Cetak deret Fibonacci
 	
System.out.println("Fibonacci Series upto " + Batas);
for(int i=0; i< Batas; i++){
 	
System.out.print(series[i] + " ");
}
}
}
	public class FiboFibonacci {
	public static void main(String[] args) {
	int Batas = 20;
	long[] series = new long[Batas];
	series[0] = 0;
	series[1] = 1;
	for(int i=2; i < Batas; i++){
	series[i] = series[i-1] + series[i-2];
	}
	System.out.println("" + Batas);
	for(int i=0; i< Batas; i++){
	System.out.print(series[i] + " ");
	}
	}
	}

Proses selanjutnya adalah proses tokenisasi adalah tahap pengelompokan statement dari kode program yang dikelompokan menjadi 5 bagian yaitu pengelompokan berupa *Header*, *Begin Class*, Deklarasi, *Begin Method*, dan Isi *Method*. Tujuan dari proses tokenisasi disini adalah untuk mengelompokkan program java yang telah dilakukan proses preporcessing kedalam bagian-bagian tertentu. Kelima bagian tersebut akan mengenali bagian-bagian dalam program java, yaitu header, class, deklarasi variabel, method dan isi dari method. Setelah proses tokensasi, maka proses selanjutnya ada implementasi algoritma greedy string tilling untuk mengetahui besarnya plagiat dari suatu program java. Algoritma greedy string tilling akan menghitung kesamaan atau plagiasi dari progam java berdasarkan jumlah masukkan kata (string) yang diketahui tingkat kesamaannya. Semakin kecil nilai string yang dimasukkan (length) makan semakin besar tingkat kesamaannya, hal ini disebabkan karena setiap kata atau string dicek satu

persatu dan jika dilihat dari sebuah program java maka pasti akan mengalami banyak kesamaan, tingkat kesamaan yang dapat digunakan untuk mengecek tingkat plagiasi dengan menggunakan algoritma GST adalah length=2 atau length=3.

Gambar 2. Uji Deteksi Tingkat Plagiasi dengan Algoritma Greedy String Tiling

HASIL DAN PEMBAHASAN

Pada penelitian ini, peneliti telah mampu mengembangkan aplikasi pendeteksian tingkat plagiasi program java. Aplikasi yang dibangun mengimplemtasikan algoritma greedy string tilling dalam proses deteksi tingkat kesamaan program satu dengan program lainnya. Peneliti melakukan proses deteksi pada program hasil praktikum pemrograman berorientasi obyek sebanyak 3 project praktikum dan 30 peserta praktikum. Proses deteksi dilakukan peneliti dengan tiga jenis panjang (length) yaitu length=1, length=2 dan length=3. Dari hasil uji didapatkan bahwa untuk program 1 dengan tingkat kesulitan adalah rendah dan dengan length=1 maka rata-rata tingkat plagiarisme mahasiswa adalah sebesar 95%, untuk program 1 dengan length=2 maka rata-rata tingkat plagiarisme adalah 73%, sendangkan untuk program 1 dengan length=3 maka tingkat plagiarisme adalah 33%. Untuk program 2 dengan length=1 maka tingkat plagiarisme adalah 97%, untuk program 2 dengan length=2 maka tingkat plagiarisme adalah 73 %, untuk program 3 dengan length=3 maka tingkat plagiarisme adalah 24%. Untuk program 3 dengan length=1 maka tingkat plagiarisme adalah 88%, untuk program 3 dengan length=2 maka tingkat plagiarisme adalah 63%, untuk program 3 dengan length=3 maka tingkat plagiarisme adalah 21%.

KESIMPULAN

Dari hasil penelitian yang dilakukan oleh peneliti didapatkan kesimpulan antara lain adalah :

1. Peneliti telah mampu mengembangkan aplikasi deteksi tingkat plagiarisme program java berbasiskan desktop.
2. Peneliti telah mengimplementasikan metode greedy string tiling dalam proses pendeteksian program java.
3. Uji aplikasi dilakukan pada program praktikum pemrograman berorientasi obyek sebanyak 3 buah program dan jumlah responden 30 peserta praktikum dan didapatkan bahwa untuk length=1 pada 3 buah program terjadi tingkat kesamaan lebih dari 85% atau tingkat berat, untuk length=2 terjadi tingkat kesamaan antara 63%-73% dan untuk length=3 terjadi tingkat kesamaan antara 21% - 33%.

DAFTAR PUSTAKA

[1] R. D. P. Suci, "Plagiarisme Dalam Pendidikan," 22-Aug-2013. [Online]. Available: http://www.kompasiana.com/resitadyah/plagiarisme-dalam-pendidikan_5528c3f76ea8340f3f8b45b8. [Accessed: 08-Jun-2017].

[2] G. K. S. Al-Shaibani, O. H. A. Mahfoodh, and F. M. Husain, "A Qualitative Investigation into the Understanding of Plagiarism in a Malaysian Research University," *J. Appl. Linguist. Lang. Res.*, vol. 3, no. 7, pp. 337–352, 2016.

[3] A. Rachman, S. Rochimah, and D. Sunaryono, "Optimization Comment and Function Detection On Open Source Programming Language Using Regex," *Int. Semin. Electr. Inform. Its Educ. Univ. Negeri Malang*, 2013.

[4] M. A. Mehmood, A. Mahmood, M. N. A. Khan, and S. Khatoon, "A scenario-based distributed testing model for software applications," *Int. J. Adv. Appl. Sci.*, vol. 3, no. 10, pp. 64–71, 2016.

[5] L. Nikhil, P. Raja, K. Yeshwanth, and N. J. K. Naidu, "SMALL DATA CHALLENGE: HOW TO USE OTHER PEOPLE'S DATA WHEN YOU DON'T HAVE YOUR OWN - SOFTWARE REUSABILITY," *Int. J. Adv. Res. Comput. Sci.*, vol. 8, no. 3, pp. 1193–1198, 2017.

[6] A. L. Gaunt *et al.*, "TERPRET: A Probabilistic Programming Language for Program Induction," *1st Workshop Neural Abstr. Mach. Program Induction NAMPI*, Dec. 2016.

[7] C. Buckler, "What's the Best Programming Language to Learn in 2017?," *Sitepoint*, 18-Jan-2017. [Online]. Available: https://www.sitepoint.com/whats-the-best-programming-language-to-learn-in-2017/. [Accessed: 08-Jun-2017].

[8] F. S. Rabbani and O. Karnalim, "Detecting Source Code Plagiarism on .NET Programming Languages using Low-level Representation and Adaptive Local Alignment," *J. Inf. Organ. Sci.*, vol. 41, no. 1, pp. 105–1123, 2017.

[9] S. Tang, L. Zou, and X. Liao, "A Research on Online Judge Technology Based on MOOC Platform," *Int. Conf. Inf. Eng. Commun. Technol. IECT 2016*, 2016.

[10] U. Pado, "Get Semantic With Me! The Usefulness of Different Feature Types for Short-Answer Grading," *26th Int. Conf. Comput. Linguist. Tech. Pap.*, pp. 2186–2195, Dec. 2016.

[11] A. Saini, A. Bahl, S. Kumari, and S. Mitali, "Plagiarism Checker: Text Mining," *Int. J. Comput. Appl.*, vol. 134, no. 3, pp. 8–11, Jan. 2016.

[12] W. Zhuang and E. Chang, "Neobility at SemEval-2017 Task 1: An Attention-based Sentence Similarity Model," *arXiv*, Mar. 2017.

[13] S. Sastroasmoro, "Beberapa Catatan Tentang Plagiarisme," *Majalan Kedokteran Indonesia*, vol. 57, no. 8, pp. 239–244, Agustus-2007.

[14] S. Ronarumata H, "Tingkat Plagiarisme pada Skripsi Mahasiswa Program Studi Ilmu Perpustakaan Lulusan Tahun 2015 berdasarkan Plagiarism Checker X Scanner." Universitas Sumatera Utara, 2016.

[15] M. Mohtarami *et al.*, "Neural-based Approaches for Ranking in Community Question Answering," *Proc. SemEval-2016*, pp. 828–835, Jun. 2016.

[16] A. Rachman, A. B. Untoro, and Mulyono, "Deteksi Komentar dan Fungsi pada Bahasa Pemrograman Berbasiskan Sumber Terbuka," *14th Semin. Intell. Tecehnology Its Appl.*, 2013.